In the Claims

The following listing of the claims replaces all previous listings.

1. (Currently Amended) A computing system having a mass storage device and a system timer for obtaining benchmark timing for a portion of an application program execution, the application program having permanently inserted performance markers, the computing system comprising:

a mass storage system;

an init module for determining if ~~the~~ timestamp data is to be collected during ~~the~~ operation of the application program;

a performance marker module for obtaining and storing the timestamp data for later retrieval at predefined points corresponding to the permanently inserted performance markers;

an uninit module for formatting and storing the obtained timestamp data into a data file within the mass storage device that permits retrieval after ~~the~~ termination of the application program; and

a performance benchmark data post processing module for determining the benchmark timing from two or more timestamp data entries;

wherein

the init module is executed before any timestamp data is collected;

the performance marker module is executed each time benchmark timestamp data and overhead timestamp data is to be collected;

the uninit module is executed after all timestamp data desired has been collected; and

the performance benchmark data post processing module determines the benchmark timing from timestamp entries stored within the data file.

2. (Original) The computing system according to claim 1, wherein the init module determines if timestamp data is to be collected.

3. (Currently Amended) The computing system according to claim 2, wherein the init module makes the determination that timestamp data is to be collected by checking for ~~the~~ existence of an identification key within a system registry;

the identification key uniquely identifying ~~the~~ processing modules to be used to collect, format, and store ~~the~~ run-time internal state data to be collected.

4. (Currently Amended) The computing system according to claim 3, wherein the timestamp data comprises a timer count value obtained from ~~the~~ a system timer.

5. (Original) The computing system according to claim 2, wherein the performance marker module collects timestamp data only if the init module has determined that the timestamp data is to be collected.

6. (Original) The computing system according to claim 5, wherein the performance marker module generates a benchmark data record containing a benchmark timestamp data value each time the performance marker module is executed.

7. (Currently Amended) The computing system according to claim 6, wherein the benchmark data record further ~~containing~~ contains an overhead timestamp data value each time the performance marker module is executed.

8. (Currently Amended) The computing system according to claim 7, wherein the performance marker module stores the benchmark data records within a data memory block within ~~the~~ processing modules identified by an identification key within a system registry.

9. (Original) The computing system according to claim 8, wherein the uninit module retrieves the benchmark data records from the data memory block for transfer to the data file on the mass storage device.

10. (Currently Amended) The computing system according to claim 9, wherein the performance benchmark data post processing module determines the benchmark timing from a difference between two benchmark timestamp data entries stored within the data file.

11. (Currently Amended) The computing system according to claim 10, wherein the performance benchmark data post processing module determines the benchmark timing by subtracting an estimate for ~~the~~ total overhead processing from ~~the~~ a difference between two benchmark timestamp data entries stored within the data file.

12. (Currently Amended) The computing system according to claim 11, wherein the estimate for the total overhead processing is determined by totaling ~~the~~ a difference between ~~the~~ an overhead timestamp value and ~~the~~ a benchmark timestamp value for all code markers between the two benchmark timestamp entries used to determine the benchmark timing.

13. (Currently Amended) A method for obtaining benchmark timing for a portion of an application program execution, the method comprising:

permanently inserting one or more code markers into the application program at locations within the application program corresponding to ~~the~~ a point at which benchmark timing data is desired;

determining if benchmark timing data is to be collected at each code marker by checking for ~~the~~ existence of processing modules identified by an identification key within a system registry;

if benchmark timing data is to be collected at each code marker:

generating a benchmark data record containing the collected benchmark timing data each time the code markers are reached;

storing the benchmark data records within a data memory block within the processing modules identified by the identification key within the system registry;

retrieving the benchmark data records from the data memory block for transfer to a mass storage device once all ~~of the~~ run-time internal state data has been collected; and

processing the benchmark data records stored within the mass storage device to determine the benchmark timing defined between two benchmark data records.

14. (Currently Amended) The method according to claim 13, wherein the benchmark timing is determined from a difference between two benchmark timestamp data entries stored within the data file.

15. (Currently Amended) The method according to claim 14, wherein the benchmark timing is determined by subtracting an estimate for ~~the~~ total overhead processing from ~~the~~ a difference between two benchmark timestamp data entries stored within the data file.

16. (Currently Amended) The method according to claim 15, wherein the estimate for the total overhead processing is determined by totaling ~~the~~ a difference between an overhead timestamp value and a benchmark timestamp value for all code markers between the two benchmark timestamp entries used to determine the benchmark timing.

17. (Original) The method according to claim 16, wherein

the benchmark timestamp value is obtained from a system timer immediately after a code marker is reached;

the overhead timestamp value is obtained from the system timer immediately before processing returns to the application program from performance marker processing.

18. (Currently Amended) A computer data product readable by a computing system and encoding a computer program of instructions for executing a computer process for obtaining run-time internal state data within an application program, said computer process comprising the steps of:

permanently inserting one or more code markers into the application program at locations within the application program corresponding to ~~the~~ a point at which benchmark timing data is desired;

~~Determining~~ determining if benchmark timing data is to be collected at each code marker by checking for ~~the~~ existence of processing modules identified by an identification key within a system registry;

if benchmark timing data is to be collected at each code marker:

generating a benchmark data record containing the collected benchmark timing data each time the code markers are reached;

storing the benchmark data records within a data memory block within the processing modules identified by the identification key within the system registry;

retrieving the benchmark data records from the data memory block for transfer to a mass storage device once all of the run-time internal state data has been collected; and

processing the benchmark data records stored within the mass storage device to determine the benchmark timing defined between two benchmark data records.

19. (Original) The computer data product according to claim 18, wherein the determining step makes the determination that benchmark timing data is to be collected by checking for the existence of an identification key within a system registry;

the identification key uniquely identifies the processing modules to be used to collect, format, and store the run-time internal state data to be collected.

20. (Original) The computer data product according to claim 19, wherein the determining step further makes the determination that benchmark timing data is to be collected by checking for the existence of processing modules identified by the identification key within the system registry

21. (Original) The computer data product according to claim 19, wherein the data memory block is within processing modules identified by the identification key within the system registry.

22. (Currently Amended) The computer data product according to claim 21, wherein the benchmark timing <u>is determined</u> from <u>a</u> difference between two benchmark timestamp data entries stored within the data file.

23. (Currently Amended) The computer data product according to claim 22, wherein the benchmark timing is determined by subtracting an estimate for ~~the~~ total overhead processing from ~~the~~ <u>a</u> difference between two benchmark timestamp data entries stored within the data file.

24. (Currently Amended) The computer data product according to claim 23, wherein the estimate for the total overhead processing is determined by totaling ~~the~~ a difference between an overhead timestamp value and a benchmark timestamp value for all code markers between the two benchmark timestamp entries used to determine the benchmark timing.

25. (Currently Amended) The computer data product according to claim 24, wherein

~~the~~ a benchmark timestamp value is obtained from a system timer immediately after a code marker is reached; and

~~the~~ an overhead timestamp value is obtained from the system timer immediately before processing returns to the application program from performance marker processing.

26. (Original) The computer data product according to claim 19, wherein the computer data product comprises a computer readable storage medium readable by a computer upon which encoded instructions used to implement the computer process are stored.

27. (Original) The computer data product according to claim 19, wherein the computer data product comprises a propagated signal on a carrier detectable by a computing system and encoding a computer program of instructions for executing the computer process.